College of Information Science and Technology

Case-Based Reasoning Approach to Reuse of Experiential Knowledge in Software Measurement Programs

Christiane Gresse von Wangenheim[1], Alexandre Moraes Ramos[1], Klaus-Dieter Althoff[2], Ricardo M. Barcia[1], Rosina Weber[1], Alejandro Martins[1]

[1] Universidade Federal de Santa Catarina - Production Engineering
Florianópolis, Brazil
{gresse,amr,rbarcia,rosina,martins}@eps.ufsc.br

[2] Fraunhofer Institute for Experimental Software Engineering
Kaiserslautern, Germany
althoff@iese.fhg.de

For the successful application of innovative software engineering technologies in industry, the technologies have to evolve incrementally based on continuous feedback from practice. Experiences about their practical application have to be systematically collected and stored in corporate memories and reused in future software projects. This promotes the sharing of experiences across individuals and projects, the formulation of best practices and facilitates the successful application of tailored technologies in practice. This paper presents a case-based reasoning approach for capturing and reusing experiential knowledge on software measurement programs in industry. A representation structure for experiential measurement knowledge is described in detail and knowledge retrieval and acquisition techniques are presented.

1 Introduction

For the improvement of quality and productivity in software organizations, many software engineering technologies have been created during the last years. These technologies, in general, provide an explicit conceptual representation of the tasks to be performed. This representation is convenient for summarizing and communicating complex task knowledge. However, while transferring innovative software engineering technologies into industry, they have to be tailored to the specific characteristics and needs of a particular organization.Through continuous learning based on feedback from their application in practice, these technologies have to be developed in an incremental and evolutionary manner.Therefore, the technology representations are likely to be simplified, often do not include aspects of their application circumstances and knowledge on how to use these technologies in practice.

Recent studies [CM96] have proposed that “experiential knowledge” [Nor93], in form of past memories, is an additional important source of knowledge which contributes to learning. Experiential knowledge guides responding new situations based on similar past experiences. Thus, accompanying technologies by experiential knowledge, describing how the tasks have to be performed while taking into account specific goals and characteristics of a particular software project will substantially facilitate

their future application in practice. Reusing experiential knowledge can prevent the repetition of past failures and guide the solution of actually occurred problems. A decreased number of problems and their efficient solution will result in cost and time savings. Furthermore, the creation of organization-specific software competencies promotes the wide-spread effective use of innovative technologies in practice. Continuous feedback from their application helps the systematic enhancement and tailoring of software engineering technologies to better meet practical needs.

In order to operationalize this "learning from measurement experiences" in industrial environments, corporate memories for the systematic acquisition and the organization wide communication of this experiential knowledge have to be built [BM96,KS96, BCR94]. As a logical and physical structure for the continuous build-up of software know-how in an organization, the *Experience Factory* (EF) approach [BCR94] (see Figure 1) has been proven to be a successful solution. The experience factory approach introduces an infrastructure for analyzing and synthesizing all kinds of experiences, acting as a repository for those, and supplying these experiences to projects on demand. The main problems concerning the operationalization of the EF in practice are to capture experiences, to represent and store knowledge in a reusable form, and reuse efficiently and effectively this knowledge in future software projects. Newly gathered experiences have to be continuously acquired and integrated into the available knowledge. *Case-based reasoning* [AP94] appears to be the optimal approach [ABG98,GAB98,Hen97] for the operationalization of an experience base in practice. The major advantages of CBR in this context are the similarity-based retrieval for primarily experiential knowledge and its continuous incremental learning as an integrated part of the reuse process.

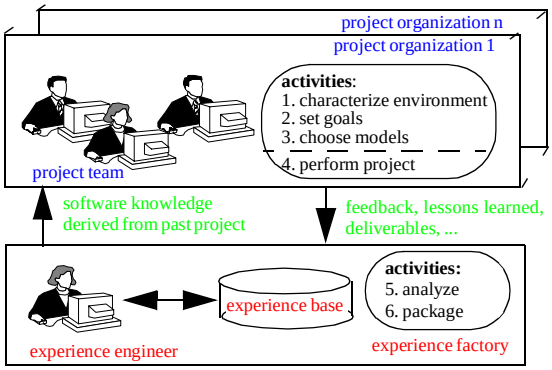

Figure 1: Experience Factory [BCR94]

We demonstrate our approach by using the Goal/Question/Metric Paradigm (GQM) [BDR97, BW84] as an innovative technology for software engineering measurement. GQM is a goal-oriented measurement approach, which helps defining and implementing operational and measurable software improvement goals. It has been successfully applied in several companies [CEM96,BCG92], such as NASA-SEL, BOSCH, Digital, and Schlumberger. Since it is an intellectually complex, resource-consuming task which requires experienced people, the availability of experiential measurement knowledge is expected to significantly contribute to the improvement of the creative process of planning measurement programs and lead to substantial effort reductions [GB97].

In this article we focus on experiential knowledge wrt. GQM-based measurement pro-

grams. A short introduction on GQM-based measurement and scenarios, illustrating the reuse potential, are given in Section 2. The case-based approach is presented in Section 3, addressing the representation of experiential measurement knowledge and techniques for the retrieval and acquisition of experiential measurement knowledge. Conclusions and future research directions are discussed in Section 4.

2 Application Domain: Software Engineering

The Goal/Question/Metric (GQM) approach is a technology for goal-oriented measurement in software projects. In GQM programs, the analysis task of measurement is specified precisely and explicitly by a detailed measurement goal, called GQM goal. Relevant measures are derived in a top-down fashion based on the goals via a set of questions and quality models. This refinement is precisely documented in a GQM plan, providing an explicit rationale for the selection of the underlying measures.The data collected is interpreted in a bottom-up fashion in the context of the GQM goal, questions and models, considering the limitations and assumptions underlying each measure. More information on GQM can be found in [BDR97,GB97,GHW95, BW84]. Here, the objective is to facilitate the establishment of measurement programs in practice by reusing experiential knowledge. The following scenarios illustrate how this knowledge can be used in order to support the GQM approach.

Scenario 1: Warning for potential failures

Tasks of the GQM process are complex and consequently error-prone to a high degree. Several critical problems can occur during its application in practice. Many problems could even be prevented, if their potential would be known in advance by the measurement responsible. Therefore, an overview on all experiences available on problems of a particular GQM task are provided to the measurement responsible before its execution. For example, during the development of the GQM plan, interviews are performed in order to acquire all relevant information wrt. the GQM goal. Before starting these interviews, the interviewer can request all experiential knowledge available on problems which occurred during this task in past measurement programs. For example: *The interviewee did not provide any information, because s/he did not know the objectives of the measurement program and which information was expected from her/him.* Knowing about problems occurred during this task in the past, will sensitize the responsible for potential problems, trying to prevent their repetition. Sometimes, problems which occur in subsequent phases of the measurement program, are due to failures during earlier phases. For example: *The development of the GQM plan was complicate, because only incomplete knowledge had been acquired during the interviews, e.g. the classification categories (low,medium,high) of the context factor "experience of developers" had not been defined. Consequently, additional follow-up interviews were necessary to precisely define the classification categories and their semantics, e.g., level of experience is high, if the developer works for more than 2 years in the development of telecommunication systems, before the development of the*

GQM plan could be continued. Providing a set of all experiential knowledge available on problems which were originated in the GQM task of interest, the measurement responsible will be aware of potential problems in advance. These experiences, providing knowledge beyond the scope of an individual or project will promote organizational learning concerning the application of software engineering technologies.

Scenario 2: Guiding the solution of a problem

When actually a problem occurs during the execution of a GQM task, its solution can be guided by experiential knowledge about similar problems and how they have been solved in past measurement programs. An example of a problem occurring during data collection is: *In the current measurement program invalid data on effort spent on software activities (e.g., hours spent on testing, hours spent on repair faults) has been collected by the project personnel.* Based on context characteristics of the problem situation (e.g. organization xyz) and the problem description (see above) relevant experiential knowledge is retrieved from the experience base and provided to the measurement responsible. Besides the context characteristics and the problem description, also the cause of the problem is important for the selection of an adequate solution strategy. Therefore, the causes of past problems, which have been explicitly captured in the past, are provided to the user. Based on these information s/he can explore the suggested reuse candidates in order to select the ones which fit best the actual needs and characteristics. For example, the cause of a similar problem in the experience base was: *Due to the weekly collection of effort data, it was difficult for the data collectors to reconstruct the time they had spent on particular activities each day.* If the actual problem was caused by a similar reason, knowledge on how the problem was solved in the past, e.g. *effort data was collected daily,* can guide the successful solution of the actual problem. An explicit description on the outcome of the solution applied on the past problem, can further indicate what worked and what did not in the particular environment, e.g. *then valid data was collected and the problem was successfully solved*. Storing also experiences regarding failed attempts to solve problems, provides additional information on potential failures aiming at their prevention in the future, e.g., *the incompleteness of the effort data collected increased, due to an increased data collection overhead through their daily collection. Furthermore the motivation and willingness of the data collectors decreased considerably.*

3 Case-Based Reasoning Approach to Capturing and Reusing Experiential Measurement Knowledge

For the experience-based support of GQM measurement programs, a case-based reasoning approach for the operationalization of experience bases for the capturing and reuse of measurement competencies in industrial environments is presented. Here, we focus on experiential knowledge gathered in individual past GQM-based measurement programs in practice. In the experience base (GQM-EKB)[1], experiential knowledge on GQM measurement are stored as cases (GQM-PSE)[2], stating the problem oc-

curred and the adopted solution strategy in the past. During the performance of new GQM measurement programs, GQM-PSEs are retrieved from the experience base, facilitating measurement. Based on context characteristics of the actual situation, e.g. name of organization, adequate cases with similar characteristics are retrieved from the experience base and provided to the user as reuse candidates. Through an interactive browsing and navigation system the user can explore the retrieved cases, select the most appropriate one, and if necessary, adapt the selected case appropriately to meet the specific needs of the actual situation. Since the GQM-EKB is used as a communication medium to share experiences organization wide, the retrieval and reuse of cases is emphasized, rather than their automated adaptation to specific characteristics of the actual situation. Integrated into the problem solving process is the acquisition of new experiential knowledge. Each time past experiences are reused in order to solve an actually occurred problem, new experiential knowledge is captured and available for problem solving.

In the following sections, we describe the representation of the experiential measurement knowledge, its retrieval and acquisition in detail.

3.1 Representation of Experiential Measurement Knowledge

As an important source for guiding the application of the GQM approach in practice, experiential measurement knowledge is captured in the GQM-EKB. Due to the specific nature of experiential knowledge, which supports the handling of exceptions, it is captured by stating an occurred problem during the performance of a GQM program and its solutions experienced in past software projects. Each case of the experience base is related to a specific problem. A GQM-PSE case includes a description of the problem, the adopted solution and information about the resulted outcome. In order to facilitate effective retrieval and provide detailed guidance for the acquisition of GQM experiences, these basic parts are refined into detailed dimensions and associated by a context description in order to allow the identification of relevant experiences in a particular environment. In addition, free-text descriptions (comments) are captured for each basic part in order to guarantee the comprehensive representation of the experiences beyond the defined dimensions. Basically, GQM-PSE consist of the following dimensions:

- **Context Description.** The organizational and project-specific context from which the experience originates is described (e.g. name of organization, programming language, application domain). In order to keep the context description minimal, only characteristics which are relevant to the particular GQM-PSE are listed. For example, the type of software (e.g. embedded software) might be irrelevant to a problem

1. This particular instantiation of an experience base is described by GQM-Experiential Knowledge Base (GQM-EKB).
2. Such a case describing a problem occurred during a GQM task and its solution is described by GQM-Problem Solution Experiences (GQM-PSE).

concerning the validity of data collection. Whereas, the duration of the software project or the size of project team might have an impact on the causes of the problem.

- **Problem**. The problem occurred during the GQM program is described.
- **Cause(s) of Problem.** The cause of the problem is explicitly described, if known. The objective is to prevent the repetition of potential problems in future measurement programs based on explicit knowledge on the causes originating the problems. As a problem can be caused through the interaction of several factors, for each cause detailed information is stated.
- **Solution.** The solution applied to solve the problem is described. Its guides coping with new problems while reusing past solution strategies in future measurement programs.
- **Outcome.** The resulted outcome of the solution applied is described in order to anticipate the expected outcome in future reuses of this case. Beside capturing successfully solved cases, also cases describing a failed tentative to solve a problem, are captured. These cases point out solutions which might potentially fail, when applied to solve the particular problem.
- **Basic information.** In order to support the appropriate usage of the available GQM-PSE and an easy and rapid selection of relevant ones, basic information on each GQM-PSE is provided:

The table below presents in detail the structure of GQM-PSE:

Case Structure	
Context Description	
Context characteristics	All relevant context characteristics are listed in form of attribute-value pairs[a]. For example, *((name of organization, xyz), (size of project team, 30 developers)).*
Comment	Additional information on the context description are stated as free-text.
Problem	
Problem description	The object affected by the problem and its state causing the problem are explicitly stated. The affected object can be any process, product or resource model or instance (e.g. data collection, effort data, tester). The state is described by listing the relevant attribute(s) of the affected object and its value[b]. For example, if invalid effort data is collected by project personnel, this can be described by *effort data: (validity, low)*.
Problem object type	The type of the objects affected by the problem is stated explicitly in order to facilitate retrieval. The objects are classified into processes, products or resources.
Problem task	The task in which the problem occurred is stated, e.g. *data collection.*
Problem roles	Roles of the software organization involved in the problem are listed, e.g., *developer*.
Goal unattained	The goal of the respective GQM task which has not been attained because of the problem is stated, e.g. *complete collection of valid data.*
Comment	Any additional information or comment on the problem is stated.
Cause(s) of Problem	
Cause description	The cause is described by the respective object and its state. For example, if the problem was caused due to the weekly collection, this is represented by *effort data collection procedure: (point in time, weekly).*
Cause explanation	An explanation for each cause is provided in order to explain the relation between the problem and the stated cause. For example, *after one week it is difficult to remember the correct amount of effort spent.*
Cause task	The task causing the actual problem, which can be different from the task of problem occurrence, is identified, e.g. *development of measurement procedures*.
Cause role(s)	Roles of the organization involved in causing the problem are stated, e.g., *developer*.

Constraint(s)	Constraints wrt. the software project which influenced the problem, e.g., wrt. the availability of resources, effort, or duration, are stated. For example, *keep collection overhead less than 1% of total project effort.*
Comment	Any additional information or comment on the cause is added as free-text.
Solution	
Solution description	The solution is described by stating the modified, added or deleted object(s) and its state. For example, if the collection procedure has been modified to daily collection, this is described by *effort data collection procedure: (point in time, daily)*.
Comment	Any additional information or comment on the solution is stated.
Justification	The solution is justified, focusing on the interdependencies between the cause, its explanation and the applied solution, e.g., *collection effort data in shorter time periods will reduce invalidity of data due to people forgetting details over time.* The justification allows the evaluation of the appropriateness of a past solution in the actual situation, while it provides an explicit rationale for its selection.
Outcome	
Outcome description	The results of the solution applied are described. It describes the state of the object stated in the problem description after the application of the solution. If, in addition, the state of other relevant objects changed due to the solution, e.g. caused new problems, these objects and their states are added. For example, if after the modification of the collection procedure, valid data is collected, this is stated by *effort data: (validity, high).*
Outcome assessment	The assessment states explicitly if the problem was successfully solved by the solution or failed, e.g. *solution was successful.*
Comment	Any additional information or comment on the outcome is described.
Failure explanation	If the applied solution failed to solve the problem, an explanation is given on why the goal of the respective task was still not achieved. For example, *because of a considerable increased effort due to the daily collection, the data collectors did not submit all required data*.
Next PSE	If the applied solution failed to solve the problem, the next attempt to solve the problem, stored as a new case in the experience base, is referenced, e.g. *case_43*.
Basic information	
Viewpoint	The role from which the knowledge was acquired is stated, e.g., *measurement expert*.
Representativeness	The representative of the GQM-PSE is given in terms of the number of individual software projects from which it was derived. For example, once captured from one software project a GQM-PSE can be confirmed in other projects, when it is reused which increases its representativeness.
Adm. information	such as creating date, ownership, access rights.

a. Attribute-value pairs are notated by (attribute, value).
b. An object and its state are notated by object: (attribute, value).

Beside the representation of explicit examples of problems and their solutions from individual projects, the representation of general domain knowledge, e.g., taxonomies and glossaries, can further facilitate the acquisition and reuse of experiential knowledge. Taxonomies represent ordered arrangements of entities according to their presumed relationships, e.g., a hierarchical taxonomy of organizational units (see Figure 2). Glossaries define organization-specific terminology and basic concepts. This domain knowledge can guide and direct the appropriate specification of relevant objects, e.g. roles, tasks. Glossaries support the adequate use of terms, their consistency across an organization, and prevent misunderstandings and communication problems between different roles, e.g., developers and senior managers.

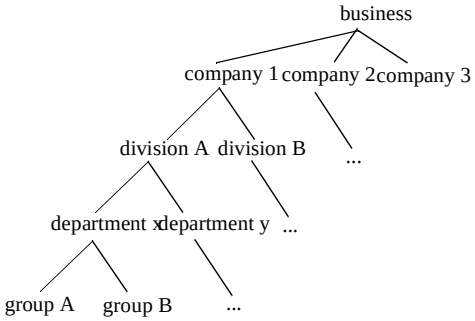

Figure 2: Organization taxonomy

3.2 Retrieval of Experiential Measurement Knowledge

The establishment of GQM measurement programs in practice is facilitated by reusing experiential knowledge. Each task of a GQM measurement program is supported by providing experiences from past measurement programs on request in order to anticipate potential problems or to solve existing ones. Based on initially given characteristics of the actual situation, similar cases are retrieved from the experience base and suggested to the user as reuse candidates. The user can explore these reuse candidates through browsing and navigation and concentrate on the ones which fit best the current needs. Depending on the objective, to prevent failures or to solve an existing problem, two applications are identified.

Application1: Warning for potential failures

Before starting a GQM task, the user can request an overview on failures occurred in past measurement programs, which were originated in this particular task (see section 2/scenario 1). The system guides the elicitation of relevant context characteristics of the actual project, e.g. name of the department, and the GQM task of interest, e.g., data collection, by questioning the user. Relevant reuse candidates are searched by exact matching of the GQM task of current interest and the Cause task of the case available in the experience base. Reuse candidates with similar context characteristics are retrieved under consideration of the characteristics describing the actual context given by the user. For example, given a specific department as organizational scope of interest, experiences gathered in this particular department are more similar to the current situation, than experiences gathered in other departments or even across companies. Since, the objective of this application is to provide an overview on all potential failures originated in the GQM task of current interest, the following information is extracted of the retrieved cases and provided to the user:

- context descriptions and basic information to allow the user to examine the validity of the proposed case in the actual situation,
- detailed descriptions of causes of problems, pointing out possible failures during the task of interest, which may cause a problem during this or a subsequent task,
- detailed descriptions of the problems occurred, anticipating what could be provoked by the failure.

Further information on the retrieved GQM-PSEs, or other GQM-PSEs available in the experience base can be explored by the user via browsing and navigation along references between individual cases.

Application2: Guidance of solution of problems

When a specific problems occurs during the GQM process, the user can request help to guide its solution by reusing solutions of past, similar problems (see section 2/scenario 2). Based on a description of the actual problem, the task when the problem occurred and specified context characteristics, a set of similar problems is retrieved from the experience base. A set of adequate reuse candidates is provided to the user, show-

ing the following information:

- context description, basic information, problem and its cause(s) in order to allow the user to evaluate in detail the validity of the suggested case in the current situation,
- solution in order to guide the solution of the actual problem, by transferring and, if necessary adapting the past solution to the actual situation. The justification for the application of the solution allows the user to evaluate the appropriateness of the suggested solution in the actual situation.
- outcome in order to inform the user about the expected consequences of the application of the solution, e.g. if it is expected to solve the problem successfully or might cause other problems.

While proposing the case to the user, s/he can further explore the dimensions of the retrieved cases, e.g, additional comments, and related cases in order to make informed decision concerning the solution of the actual problem. If necessary, suggested solutions are adapted by the user to meet the current needs.

3.3 Acquisition of Experiential Measurement Knowledge

Essential for continuous improvement of software engineering technologies is their incremental evolution based on feedback from industrial applications. Consequently, the knowledge in the experience base has to be enhanced and updated each time a new measurement program is established. Each time a problem occurs during a GQM measurement programs, it is captured as a new GQM-PSE. The acquisition of new experiences is intertwined into the problem solving process. While the user requests experience-based support for the solution of an occurred problem, the initially given problem description and context characteristics used for retrieval of similar cases are in parallel captured documenting a new GQM-PSE. If cases retrieved from the GQM-EKB are used for the solution of the actual situation, the reused case serves as a basis for the further documentation of the actual situation. The description of the new case is carefully reviewed by the user and missing information is added and deviations from the reused case are modified. For example, if the same problem occurs wrt. the collection of fault data instead of effort data, the case description has to be modified wrt. the type of data collected. The acquisition of new experiences is guided through the detailed case structure, which explicitly addresses relevant dimensions. The usage of organizational domain knowledge, such as glossaries and taxonomies, further facilitates the consistent description of experiences across individuals and projects.

4 Conclusions

For the successful application and continuous evolution of software engineering technologies in practice, experiential knowledge has to be captured in corporate memories and reused in future applications. In this paper, we present a case-based reasoning approach for the operationalization of experience bases on experiential measurement knowledge. We describe the representation structure of the experiential knowledge

and techniques for the retrieval of adequate reuse candidates and the acquisition of new experiences. Currently, we are implementing a an experience base for the experience-based support of the planning of GQM measurement programs by using a CBR-tool [Tec97]. The application of the approach offers the possibility of systematic acquisition of experiential measurement knowledge in practice and, therefore, provides a basis for further research on the evolution and generalization of technologies wrt. packaging and reuse of experiential knowledge on software engineering technologies, based on feedback from its use in practice.

5 References

[ABG98] K.D. Althoff, A. Birk, C. Gresse von Wangenheim, C. Tautz. CBR for Experimental Software Engineering. In H. D. Burkhard, M. Lenz et al., eds., Case-Based Reasoning Technology from Foundations to Applications, Springer, Berlin, to appear 1998.

[AP94] A. Aamodt, E. Plaza. Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. AI Communications, vol. 17, nr. 1, 1994.

[BM96] J.M. Barr, R.V. Magaldi.Corporate Knowledge Management for the Millennium. In I. Smith and B. Faltings, ed., Advances in Case-Based Reasoning, Springer, Berlin, 1996.

[BCG92] V. R. Basili et al. The Software Engineering Laboratory-An Operational Software Experience Factory. In Proceedings of the 14th Int. Conference on Software Engineering, 1992.

[BCR94] V. R. Basili, G. Caldiera, H. D. Rombach. Experience Factory. In John J. Marciniak, editor, Encyclopedia of Software Engineering, vol.1, pp. 528-532. John Wiley & Sons, 1994.

[BDR97] L. C. Briand, C. M. Differding, H. D. Rombach. Practical Guidelines for Measurement-Based Process Improvement. Software Process Improvement and Practice, to appear 1997.

[BW84] V. R. Basili, D. M. Weiss. A Methodology for Collecting Valid Software Engineering Data. IEEE Transactions on Software Engineering, SE-10(6):728-738, 1984.

[CEM96] CEMP Consortium. Customized Establishment of Measurement Programs. Final Report, ESSI Project Nr. 10358, Germany,1996.

[CM96] JR. Cho,R.C. Mathews.Interactions Between Mental Models Used in Categorization and Experiential Knowledge of Specific Cases. The Journal of Experimental Psychology, 49A (3), 1996 .

[GB97] C. Gresse, L.C. Briand. Requirements for the Knowledge-Based Support of Software Engineering Measurement Plans. In Proceedings of the 9th Int. Conference on Software Engineering and Knowledge Engineering, Spain, 1997.

[GAB98] C. Gresse von Wangenheim, K-D. Althoff, R. M. Barcia, C. Tautz. Evaluation of Technologies for Packaging and Reuse of Software Engineering Experiences. Submitted to Int. Conference on Industrial Engineering Applications of Artificial Intelligence and Expert Systems, 1998.

[GHW95] C. Gresse, B. Hoisl, J. Wüst. A Process Model for GQM- Based Measurement. Technical Report STTI-95-04-E, Software Technology Transfer Initiative, Germany, 1995.

[Hen97] S. Henninger. Capturing and Formalizing Best Practices in a Software Development Organization. In Proceedings of the 9th Int. Conference on Software Engineering and Knowledge Engineering, Spain, 1997.

[KS96] H. Kitano, H. Shimazu. The Experience-Sharing Architecture. In D. Leake, ed., Case-Based Reasoning Experiences: Lessons Learned & Future Directions, 1996.

[Nor93] D.A. Norman. Things That Make Us Smart: Defending Human Attributes in the Age of the Machine. Reading Ma, Addison-Wesley, 1993.

[Tec97] CBR-Works. TecInno GmbH, Germany (http://www.tecinno.de/tecinno_e/ecbrwork.htm)